

BankAccount: Reading the Code

Name: _____

Date: _____

Use BankAccount.java to answer the questions below.

Part 1: Find it in the code

1. How many constructors does BankAccount have? How does Java decide which one to call?
2. List every field. For each one, say whether it is static or instance, and whether it is public or private.
3. Which fields are final? What does final mean for each one?
4. How many accessor (getter) methods are there? How many mutator (setter) methods?
5. Which method is private? Who is allowed to call it?
6. List every static method. How do you call one from outside the class?
7. Find every place in the code where a static field is used.

Part 2: Trace it

8. Compare the two constructors line by line. What is the only real difference between them? Which lines are repeated in both, and what could go wrong if someone later changes one constructor but forgets the other?
9. Suppose you create three accounts. What does BankAccount.getNumAccounts() return? What account number does the third account get?
10. An account has \$30. What happens when you call withdraw(50)? What does the method return?
11. What happens when you call deposit(-20)?

12. Account a has \$100 and account b has \$50. Someone calls `BankAccount.setInterestRate(0.05)`, then `a.addInterest()`. What are the two balances afterward?
13. What does `System.out.println(a)` print, and why don't you have to write `a.toString()`?

Part 3: Think about it

14. In the constructor, why do we write `this.owner = owner` instead of `owner = owner`?
15. There is a `getBalance()` method but no `setBalance()`. Why might that be on purpose?
16. Why is it fine for `MIN_BALANCE` to be public when `balance` is private?
17. `nickname` is public. Write one line of code, outside the class, that changes it. Why is a public field like this usually considered bad design?
18. Why can't `getNumAccounts()` return `balance`?
19. What does `.equals` compare? Two accounts both belong to "Ada" and both have \$60. Are they `.equals`? Should they be?
20. What is the difference between `a == c` and `a.equals(c)`? Describe a situation where `==` is false but `.equals` would be true.
21. Suppose that in a constructor, the account number was assigned before `numAccounts++`. What would change?
22. Can `main` call `computeInterest()` even though it's private? Can a different class call it?